FastLAS: Scalable Inductive Logic Programming, incorporating Domain-specific Optimisation Criteria

Mark Law, Alessandra Russo, Elisa Bertino,
Krysia Broda and Jorge Lobo

February 11, 2020

Inductive Logic Programming

Input: background knowledge B and examples E^+ and E^-.

Output: a hypothesis H s.t.

1. $\forall e^+ \in E^+ : B \cup H \models e^+$
2. $\forall e^- \in E^- : B \cup H \not\models e^-$.

Inductive Logic Programming

Input: background knowledge B and examples E^+ and E^-.

Output: a hypothesis H s.t.

1. $\forall e^+ \in E^+ : B \cup H \models e^+$
2. $\forall e^- \in E^- : B \cup H \not\models e^-$.

The paper contains two main contributions:

- FastLAS, an ASP-based ILP system which is scalable w.r.t. the size of the hypothesis space.
- A new approach to supporting domain specific optimisation criteria in the search.

Imperial College London

Optimisation Criteria

In some domains, false negatives are more dangerous than false positives.

Event detection

Medical diagnosis

Optimisation Criteria

In some domains, false negatives are more dangerous than false positives.

Event detection

Medical diagnosis

In others (e.g. access control), false positives may be more dangerous than false negatives.

Optimisation Criteria

In some domains, false negatives are more dangerous than false positives.

Event detection

Medical diagnosis

In others (e.g. access control), false positives may be more dangerous than false negatives.

Depending on the domain, different optimisation criteria may be required!

Optimisation Criteria in ILP

Many ILP systems aim to find the simplest solution.

- Optimal solutions of a task T minimise $\mathcal{S}_{len}(H, T) = |H|$.

Optimisation Criteria in ILP

Many ILP systems aim to find the simplest solution.

- Optimal solutions of a task T minimise $\mathcal{S}_{len}(H, T) = |H|$.
- With noise, minimise $\mathcal{S}_{len}(H, T) + \sum_{e \in UNCOV(H,T)} e_{pen}$.

Optimisation Criteria in ILP

Many ILP systems aim to find the simplest solution.

- Optimal solutions of a task T minimise $\mathcal{S}_{len}(H, T) = |H|$.
- With noise, minimise $\mathcal{S}_{len}(H, T) + \sum_{e \in UNCOV(H,T)} e_{pen}$.

Many other optimisation criteria exist:

- The number of rules in H.
- The number of ground instances of rules in H.
- The number of variables in H.
- The number of answer sets of $B \cup H$.

Optimisation Criteria in ILP

Many ILP systems aim to find the simplest solution.

- Optimal solutions of a task T minimise $\mathcal{S}_{len}(H, T) = |H|$.
- With noise, minimise $\mathcal{S}_{len}(H, T) + \sum_{e \in UNCOV(H,T)} e_{pen}$.

Many other optimisation criteria exist:

- The number of rules in H.
- The number of ground instances of rules in H.
- The number of variables in H.
- The number of answer sets of $B \cup H$.

A general *scoring function* takes as input a learning task T and a hypothesis H and returns a *score* in $\mathbb{R}_{\geq 0}$.

Optimisation Criteria in ILP

Many ILP systems aim to find the simplest solution.

- Optimal solutions of a task T minimise $\mathcal{S}_{len}(H, T) = |H|$.
- With noise, minimise $\mathcal{S}_{len}(H, T) + \sum_{e \in UNCOV(H,T)} e_{pen}$.

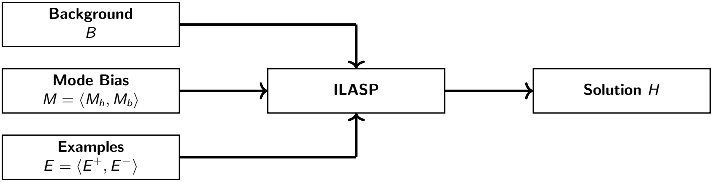
Many other optimisation criteria exist:

- The number of rules in H.
- The number of ground instances of rules in H.
- The number of variables in H.
- The number of answer sets of $B \cup H$.

FastLAS currently supports *decomposable* scoring functions. These can be evaluated on each rule independently.

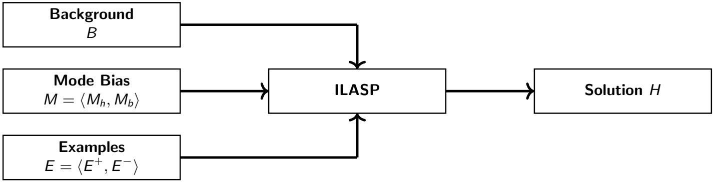
FastLAS Learning Task

The FastLAS task T is based on the ILASP (Law et al. 2015) task:



FastLAS Learning Task

The FastLAS task T is based on the ILASP (Law et al. 2015) task:

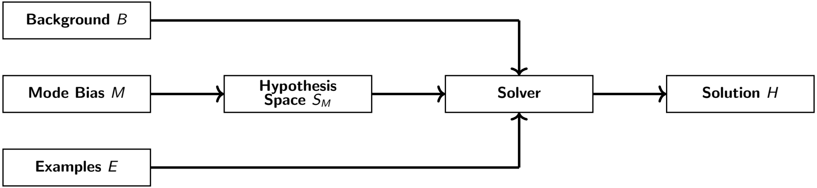


FastLAS has two major restrictions:

- All programs have a single answer set.
- No predicate in M_h occurs in M_b or any rule in T.

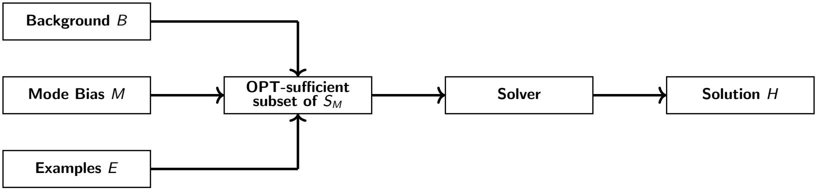
FastLAS: Overview

ILASP begins by computing the hypothesis space S_M, which contains every rule that is compatible with the mode bias M.



FastLAS: Overview

FastLAS instead begins by computing an *OPT-sufficient* subset of the hypothesis space, which is often significantly smaller than S_M.



FastLAS

Initial Construction

Generalisation

Optimisation

FastLAS

Initial Construction

Generalisation

Optimisation

Generate the set of all “maximal” rules which could be useful.

$\mathcal{C}^{+}(T)$ contains the “maximal” rules that prove something we **should** prove.
$\mathcal{C}^{-}(T)$ contains the “maximal” rules that prove something we **shouldn’t** prove.

Imperial College London

FastLAS

Initial Construction

Generalisation

Optimisation

Generate the set of all "maximal" rules which could be useful.

$\mathcal{C}^{+}(T)$ contains the "maximal" rules that prove something we **should** prove.
$\mathcal{C}^{-}(T)$ contains the "maximal" rules that prove something we **shouldn't** prove.

Example:

$$B = \emptyset \qquad M_h = \{\texttt{p}, \texttt{q}\} \qquad M_b = \left\{ \begin{array}{c} \texttt{r}, \texttt{not r}, \\ \texttt{s}, \texttt{not s} \\ \texttt{t}, \texttt{u} \end{array} \right\}$$

FastLAS

Initial Construction

Generalisation

Optimisation

Generate the set of all "maximal" rules which could be useful.

$\mathcal{C}^{+}(T)$ contains the "maximal" rules that prove something we **should** prove.
$\mathcal{C}^{-}(T)$ contains the "maximal" rules that prove something we **shouldn't** prove.

Example:

$$B = \emptyset \qquad M_h = \{\texttt{p}, \texttt{q}\} \qquad M_b = \left\{ \begin{array}{c} \texttt{r}, \texttt{not r}, \\ \texttt{s}, \texttt{not s} \\ \texttt{t}, \texttt{u} \end{array} \right\}$$

$$\text{Find } H \text{ s.t. } \left\{ \begin{array}{cc} H \models \texttt{q}, & H \not\models \texttt{p}, \\ H \cup \{\texttt{r. t.}\} \models \texttt{p}, & H \cup \{\texttt{r. t.}\} \not\models \texttt{q}, \\ H \cup \{\texttt{r. u.}\} \models \texttt{p}, & H \cup \{\texttt{r. u.}\} \not\models \texttt{q} \end{array} \right\}$$

FastLAS

Initial Construction

Generalisation

Optimisation

Generate the set of all "maximal" rules which could be useful.

$\mathcal{C}^+(T)$ contains the "maximal" rules that prove something we **should** prove.
$\mathcal{C}^-(T)$ contains the "maximal" rules that prove something we **shouldn't** prove.

Example:

$$B = \emptyset \qquad M_h = \{\texttt{p}, \texttt{q}\} \qquad M_b = \left\{ \begin{array}{c} \texttt{r}, \texttt{not r}, \\ \texttt{s}, \texttt{not s} \\ \texttt{t}, \texttt{u} \end{array} \right\}$$

$$\text{Find } H \text{ s.t.} \left\{ \begin{array}{cc} H \models \texttt{q}, & H \not\models \texttt{p}, \\ H \cup \{\texttt{r. t.}\} \models \texttt{p}, & H \cup \{\texttt{r. t.}\} \not\models \texttt{q}, \\ H \cup \{\texttt{r. u.}\} \models \texttt{p}, & H \cup \{\texttt{r. u.}\} \not\models \texttt{q} \end{array} \right\}$$

$$\mathcal{C}^+(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \end{array} \right\} \quad \mathcal{C}^-(T) = \left\{ \begin{array}{l} \texttt{p :- not r, not s.} \\ \texttt{q :- r, not s, t.} \\ \texttt{q :- r, not s, u.} \end{array} \right\}$$

Imperial College London

FastLAS

Initial Construction

Generalisation

Optimisation

Compute "maximal" generalisations.

$\mathcal{G}(T)$ contains the maximal subrules of rules in $\mathcal{C}^+(T)$ that are subrules of as many rules in $\mathcal{C}^+(T)$ as possible.

Example:

$$\mathcal{C}^+(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \end{array} \right\}$$

FastLAS

Initial Construction

Generalisation

Optimisation

Compute "maximal" generalisations.

$\mathcal{G}(T)$ contains the maximal subrules of rules in $\mathcal{C}^+(T)$ that are subrules of as many rules in $\mathcal{C}^+(T)$ as possible.

Example:

$$\mathcal{C}^+(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \end{array} \right\} \qquad \mathcal{G}(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \\ \texttt{p :- r, not s.} \end{array} \right\}$$

FastLAS

Initial Construction

Generalisation

Optimisation

Compute OPT-sufficient subset of the hypothesis space

For each $R \in \mathcal{G}(T)$, compute *one* optimal subrule of R w.r.t. $\mathcal{S}$ that is not a subrule of any rule in $\mathcal{C}^{-}(T)$.

FastLAS

Initial Construction

Generalisation

Optimisation

Compute OPT-sufficient subset of the hypothesis space

For each $R \in \mathcal{G}(T)$, compute *one* optimal subrule of R w.r.t. $\mathcal{S}$ that is not a subrule of any rule in $\mathcal{C}^{-}(T)$.

Example:

$$\mathcal{G}(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \\ \texttt{p :- r, not s.} \end{array} \right\} \quad \mathcal{C}^{-}(T) = \left\{ \begin{array}{l} \texttt{p :- not r, not s.} \\ \texttt{q :- r, not s, t.} \\ \texttt{q :- r, not s, u.} \end{array} \right\}$$

Optimise with $\mathcal{S}_{len}$:

FastLAS

Initial Construction

Generalisation

Optimisation

Compute OPT-sufficient subset of the hypothesis space

For each $R \in \mathcal{G}(T)$, compute *one* optimal subrule of R w.r.t. $\mathcal{S}$ that is not a subrule of any rule in $\mathcal{C}^{-}(T)$.

Example:

$$\mathcal{G}(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \\ \texttt{p :- r, not s.} \end{array} \right\} \quad \mathcal{C}^{-}(T) = \left\{ \begin{array}{l} \texttt{p :- not r, not s.} \\ \texttt{q :- r, not s, t.} \\ \texttt{q :- r, not s, u.} \end{array} \right\}$$

Optimise with $\mathcal{S}_{len}$:

$\texttt{q :- not r, not s.} \Rightarrow \texttt{q :- not r.}$

FastLAS

Initial Construction

Generalisation

Optimisation

Compute OPT-sufficient subset of the hypothesis space

For each $R \in \mathcal{G}(T)$, compute *one* optimal subrule of R w.r.t. $\mathcal{S}$ that is not a subrule of any rule in $\mathcal{C}^{-}(T)$.

Example:

$$\mathcal{G}(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \\ \texttt{p :- r, not s.} \end{array} \right\} \quad \mathcal{C}^{-}(T) = \left\{ \begin{array}{l} \texttt{p :- not r, not s.} \\ \texttt{q :- r, not s, t.} \\ \texttt{q :- r, not s, u.} \end{array} \right\}$$

Optimise with $\mathcal{S}_{len}$:

$$\begin{array}{lcl} \texttt{q :- not r, not s.} & \Rightarrow & \texttt{q :- not r.} \\ \texttt{p :- r, not s, t.} & \Rightarrow & \texttt{p :- r.} \end{array}$$

FastLAS

Initial Construction

Generalisation

Optimisation

Compute OPT-sufficient subset of the hypothesis space

For each $R \in \mathcal{G}(T)$, compute *one* optimal subrule of R w.r.t. $\mathcal{S}$ that is not a subrule of any rule in $\mathcal{C}^{-}(T)$.

Example:

$$\mathcal{G}(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \\ \texttt{p :- r, not s.} \end{array} \right\} \quad \mathcal{C}^{-}(T) = \left\{ \begin{array}{l} \texttt{p :- not r, not s.} \\ \texttt{q :- r, not s, t.} \\ \texttt{q :- r, not s, u.} \end{array} \right\}$$

Optimise with $\mathcal{S}_{len}$:

$$\begin{array}{lcl} \texttt{q :- not r, not s.} & \Rightarrow & \texttt{q :- not r.} \\ \texttt{p :- r, not s, t.} & \Rightarrow & \texttt{p :- r.} \\ \texttt{p :- r, not s, u.} & \Rightarrow & \texttt{p :- r.} \end{array}$$

FastLAS

Initial Construction

Generalisation

Optimisation

Compute OPT-sufficient subset of the hypothesis space

For each $R \in \mathcal{G}(T)$, compute *one* optimal subrule of R w.r.t. $\mathcal{S}$ that is not a subrule of any rule in $\mathcal{C}^{-}(T)$.

Example:

$$\mathcal{G}(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \\ \texttt{p :- r, not s.} \end{array} \right\} \quad \mathcal{C}^{-}(T) = \left\{ \begin{array}{l} \texttt{p :- not r, not s.} \\ \texttt{q :- r, not s, t.} \\ \texttt{q :- r, not s, u.} \end{array} \right\}$$

Optimise with $\mathcal{S}_{len}$:

$$\begin{array}{lcl} \texttt{q :- not r, not s.} & \Rightarrow & \texttt{q :- not r.} \\ \texttt{p :- r, not s, t.} & \Rightarrow & \texttt{p :- r.} \\ \texttt{p :- r, not s, u.} & \Rightarrow & \texttt{p :- r.} \\ \texttt{p :- r, not s.} & \Rightarrow & \texttt{p :- r.} \end{array}$$

FastLAS

Initial Construction

Generalisation

Optimisation

Compute OPT-sufficient subset of the hypothesis space

For each $R \in \mathcal{G}(T)$, compute *one* optimal subrule of R w.r.t. $\mathcal{S}$ that is not a subrule of any rule in $\mathcal{C}^{-}(T)$.

Example:

$$\mathcal{G}(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \\ \texttt{p :- r, not s.} \end{array} \right\} \quad \mathcal{C}^{-}(T) = \left\{ \begin{array}{l} \texttt{p :- not r, not s.} \\ \texttt{q :- r, not s, t.} \\ \texttt{q :- r, not s, u.} \end{array} \right\}$$

Optimise with $\mathcal{S}_{len}$:

$$\begin{array}{lcl} \texttt{q :- not r, not s.} & \Rightarrow & \texttt{q :- not r.} \\ \texttt{p :- r, not s, t.} & \Rightarrow & \texttt{p :- r.} \\ \texttt{p :- r, not s, u.} & \Rightarrow & \texttt{p :- r.} \\ \texttt{p :- r, not s.} & \Rightarrow & \texttt{p :- r.} \end{array}$$

$$\mathcal{O}(T, \mathcal{S}_{len}) = \left\{ \begin{array}{l} \texttt{q :- not r.} \\ \texttt{p :- r.} \end{array} \right\}$$

FastLAS

Initial Construction

Generalisation

Optimisation

Compute OPT-sufficient subset of the hypothesis space

For each $R \in \mathcal{G}(T)$, compute *one* optimal subrule of R w.r.t. $\mathcal{S}$ that is not a subrule of any rule in $\mathcal{C}^{-}(T)$.

Example:

$$\mathcal{G}(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \\ \texttt{p :- r, not s.} \end{array} \right\} \quad \mathcal{C}^{-}(T) = \left\{ \begin{array}{l} \texttt{p :- not r, not s.} \\ \texttt{q :- r, not s, t.} \\ \texttt{q :- r, not s, u.} \end{array} \right\}$$

Optimise with $\mathcal{S}_{len}$:

$$\begin{array}{lcl} \texttt{q :- not r, not s.} & \Rightarrow & \texttt{q :- not r.} \\ \texttt{p :- r, not s, t.} & \Rightarrow & \texttt{p :- r.} \\ \texttt{p :- r, not s, u.} & \Rightarrow & \texttt{p :- r.} \\ \texttt{p :- r, not s.} & \Rightarrow & \texttt{p :- r.} \end{array}$$

$$\mathcal{O}(T, \mathcal{S}_{len}) = \left\{ \begin{array}{l} \texttt{q :- not r.} \\ \texttt{p :- r.} \end{array} \right\}$$

$\mathcal{O}(T, \mathcal{S})$ is *OPT-sufficient*. Hence, FastLAS is guaranteed to return an optimal solution w.r.t. any decomposable scoring function!

Imperial College London

FastLAS

Initial Construction

Generalisation

Optimisation

Compute OPT-sufficient subset of the hypothesis space

For each $R \in \mathcal{G}(T)$, compute *one* optimal subrule of R w.r.t. $\mathcal{S}$ that is not a subrule of any rule in $\mathcal{C}^{-}(T)$.

Example:

$$\mathcal{G}(T) = \left\{ \begin{array}{l} \texttt{q :- not r, not s.} \\ \texttt{p :- r, not s, t.} \\ \texttt{p :- r, not s, u.} \\ \texttt{p :- r, not s.} \end{array} \right\} \quad \mathcal{C}^{-}(T) = \left\{ \begin{array}{l} \texttt{p :- not r, not s.} \\ \texttt{q :- r, not s, t.} \\ \texttt{q :- r, not s, u.} \end{array} \right\}$$

Optimise with $\mathcal{S}_{len}$:

$$\begin{array}{lcl} \texttt{q :- not r, not s.} & \Rightarrow & \texttt{q :- not r.} \\ \texttt{p :- r, not s, t.} & \Rightarrow & \texttt{p :- r.} \\ \texttt{p :- r, not s, u.} & \Rightarrow & \texttt{p :- r.} \\ \texttt{p :- r, not s.} & \Rightarrow & \texttt{p :- r.} \end{array}$$

$$\mathcal{O}(T, \mathcal{S}_{len}) = \left\{ \begin{array}{l} \texttt{q :- not r.} \\ \texttt{p :- r.} \end{array} \right\}$$

$\mathcal{O}(T, \mathcal{S})$ is *OPT-sufficient*. Hence, FastLAS is guaranteed to return an optimal solution w.r.t. any decomposable scoring function!

The full hypothesis space (computed by ILASP) contains 72 rules.

Evaluation

Sentence Chunking

In (Kazmi et al. 2017), the Inspire system was evaluated on a sentence chunking dataset (Agirre et al. 2016), which contains examples of how sentences should be *chunked*.

Sentence Chunking

In (Kazmi et al. 2017), the Inspire system was evaluated on a sentence chunking dataset (Agirre et al. 2016), which contains examples of how sentences should be *chunked*.

System	F_1	Running Time
INSPIRE	0.712	>1800s in some cases
ILASP3	0.777	1051.4s
FastLAS	0.768	4.5s

Policy Learning II

We evaluated FastLAS on access control datasets.

- We used three scoring functions, $\mathcal{S}_{len}$, $\mathcal{S}_{cov}$ and $\mathcal{S}_{uni}$, which encouraged learning progressively more general hypotheses.

Policy Learning II

We evaluated FastLAS on access control datasets.

- We used three scoring functions, $\mathcal{S}_{len}$, $\mathcal{S}_{cov}$ and $\mathcal{S}_{uni}$, which encouraged learning progressively more general hypotheses.

Learning rules for *accept*:

Scoring function	Recall	Precision
$\mathcal{S}_{len}$	0.905	0.951
$\mathcal{S}_{cov}$	0.892	0.949
$\mathcal{S}_{uni}$	0.991	0.917

Policy Learning II

We evaluated FastLAS on access control datasets.

- We used three scoring functions, $\mathcal{S}_{len}$, $\mathcal{S}_{cov}$ and $\mathcal{S}_{uni}$, which encouraged learning progressively more general hypotheses.

Learning rules for *accept*:

Scoring function	Recall	Precision
$\mathcal{S}_{len}$	0.905	0.951
$\mathcal{S}_{cov}$	0.892	0.949
$\mathcal{S}_{uni}$	0.991	0.917

Learning rules for *reject*:

Scoring function	Recall	Precision
$\mathcal{S}_{len}$	0.974	0.935
$\mathcal{S}_{cov}$	0.969	0.941
$\mathcal{S}_{uni}$	0.966	0.965

Conclusion

FastLAS is a new ASP-based ILP system.

- Far more scalable than ILASP.
- Supports domain-specific optimisation criteria, through scoring functions.

Future research directions include:

- Lifting current restrictions, to allow recursion, non-observational predicate learning, predicate invention and programs with multiple answer sets.
- Non-decomposable scoring functions.

Backup Slides

Sentence Chunking Results

100 examples:

System	F_1	Running Time
INSPIRE	0.733	–
ILASP3	0.757	210.4s
FastLAS	0.751	0.909s

500 examples:

System	F_1	Running Time
INSPIRE	0.712	–
ILASP3	0.777	1051.4s
FastLAS	0.768	4.5s

CAVIAR

The CAVIAR dataset (generated by the EC Funded CAVIAR project/IST 2001 37540) consists of manually annotated video streams.

The task is to detect events by learning initiating/terminating conditions. We compared FastLAS to OLED (Katzouris et al. 2016) and ILASP3 (Law et al. 2015) on the task of detecting people meeting.

CAVIAR Results

System	F_1	Running Time
OLED	0.792	107s
ILASP3	0.837	523.3s
FastLAS	0.907	263.8s

For CAVIAR, ILASP had a hypothesis space with 3370 rules. FastLAS had over 2^{44} rules.

CAVIAR Results

System	F_1	Running Time
OLED	0.792	107s
ILASP3	0.837	523.3s
FastLAS	0.907	263.8s

For CAVIAR, ILASP had a hypothesis space with 3370 rules. FastLAS had over 2^{44} rules.

Due to the larger search space, FastLAS achieved an F_1 score of 0.923 compared to ILASPs 0.842.

Policy Learning II

In access control logs, users do not tend to request resources which they know they cannot access, meaning negative examples do not occur frequently.

Policy Learning II

In access control logs, users do not tend to request resources which they know they cannot access, meaning negative examples do not occur frequently.

In (Cotrini et al. 2018), a new metric, *universal* F_1 was proposed, which uses a modified version of *precision* ($P = \frac{tp}{tp+fp}$), $UP = \frac{tp}{tp+fp+u}$, where $\mathtt{u}$ is the set of users who have not requested access to the resource.

Policy Learning II

In access control logs, users do not tend to request resources which they know they cannot access, meaning negative examples do not occur frequently.

In (Cotrini et al. 2018), a new metric, *universal* F_1 was proposed, which uses a modified version of *precision* ($P = \frac{tp}{tp+fp}$), $UP = \frac{tp}{tp+fp+u}$, where $\mathtt{u}$ is the set of users who have not requested access to the resource.

Method	Resource 25993	Resource 4675	Resource 75078	Resource 79092
Rhapsody	0.04	0.10	0.10	0.04
CTA	0.04	**0.12**	0.10	0.04
FastLAS: $\mathcal{S}_{len}$	0.02	0.04	0.02	0.02
FastLAS: $\mathcal{S}_{cov}$	0.02	0.04	0.02	0.02
FastLAS: $\mathcal{S}_{uni}$	0.01	0.04	0.02	0.02
FastLAS: $\mathcal{S}_{UF_1}$	**0.07**	0.10	**0.11**	**0.05**

Agirre, E., Gonzalez Agirre, A., Lopez-Gazpio, I., Maritxalar, M., Rigau Claramunt, G., and Uria, L. 2016.
Semeval-2016 task 2: Interpretable semantic textual similarity.
In *Proceedings of the Tenth International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 512–524.

Cotrini, C., Weghorn, T., and Basin, D. 2018.
Mining abac rules from sparse logs.
In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 31–46.

Katzouris, N., Artikis, A., and Paliouras, G. 2016.
Online learning of event definitions.
Theory and Practice of Logic Programming 16, 5-6, 817–833.

Kazmi, M., Schüller, P., and Saygin, Y. 2017.
Improving scalability of inductive logic programming via pruning and best-effort optimisation.
Expert Systems with Applications 87, 291–303.

Law, M., Russo, A., and Broda, K. 2015.
The ILASP system for learning answer set programs.
www.ILASP.com.