universität innsbruck


# Machine Learning for Theorem Proving

Lecture 4 (VU)

Cezary Kaliszyk

# Overview

**Last Lecture**

- Naive Bayes and k-NN

**Today**

- Syntactic methods for premise selection
- Linear Regression
- Random forests

# SInE: introduction

We will look at an algorithm that does not actually learn from the proofs of theorems, but only looks at their statements.

This algorithm that is more used in automated systems and not in interactive provers. However, the MePo relevance filter is a very similar technique in proof assistants.

# SInE: original idea by Hoder

## Basic algorithm

If symbol $s$ is $d$-relevant and appears in axiom $a$, then $a$ and all symbols in $a$ become $d + 1$-relevant.

## Problem: Common Symbols

- Simple *relevance* usually selects all axioms
- Because of common symbols, such as `subclass` or `subsumes`

```
subclass (beverage, liquid).
subclass (chair, furniture).
```

## Solution: Trigger based selection

“appears” is changed to “triggers”

## But how to know if $s$ is common?

Approximate by number of occurrences in the current problem

# SInE: Tolerance

- Only symbols with $t$-times more occurrences than the least common symbol trigger an axiom
- For $t = \infty$ this is the same as relevance

subclass(X,Y) ∧ subclass(Y,Z) → subclass(X,Z)
subclass(petrol,liquid)
¬subclass(stone,liquid)
subclass(beverage,liquid)
subclass(beer,beverage)
subclass(guinness,beer)
subclass(pilsner,beer)

? subclass(beer,liquid)

[*Hoder*]

What are the symbol occurrence numbers and when are they triggered?

# SInE: Tolerance

- Only symbols with $t$-times more occurrences than the least common symbol trigger an axiom
- For $t = \infty$ this is the same as relevance

**1:** subclass(X,Y) ∧ subclass(Y,Z) → subclass(X,Z)
subclass(petrol,liquid)
¬subclass(stone,liquid)
**2:** subclass(beverage,liquid)
**1:** subclass(beer,beverage)
subclass(guinness,beer)
subclass(pilsner,beer)

| Occ. | Symbols |
|---|---|
| 7 | subclass |
| 3 | liquid, beer |
| 2 | beverage |
| 1 | petrol, stone, guinness, pilsner |

? subclass(beer,liquid)

[*Hoder*]

# SInE: Commentary

In the example, the right table lists symbol occurrences, the left table lists the rounds in which particular axioms are triggered. In two rounds exactly the needed axioms to prove the conjecture are triggered.

The next slide shows a number of tables that show the space reduction with Sine. Mizar Is a proof assistant developed in Poland (University of Białystok). It has one of the largest databases of facts and all are in a simple logic (close to first-order logic), and as such it is a very interesting benchmark for machine learning methods. SUMO is a manual encoding of a large ontology, which aims to describe many relations in the real world using first-order logic.

The slide shows that depending on the symbol density and size of statements, different values of parameters allow reducing the size of the bases using Sine appropriately.

# SInE: Performance

Problem sizes:

| problems | axioms | atoms | predicates | functions |
|---|---|---|---|---|
| SUMO | 298,420 | 323,170 | 20 | 24,430 |
| CYC | 3,341,990 | 5,328,216 | 204,678 | 1,050,014 |
| Mizar | 44,925 | 332,143 | 2,328 | 6,115 |

# SInE: Performance

Problem sizes:

| problems | axioms | atoms | predicates | functions |
|---|---|---|---|---|
| SUMO | 298,420 | 323,170 | 20 | 24,430 |
| CYC | 3,341,990 | 5,328,216 | 204,678 | 1,050,014 |
| Mizar | 44,925 | 332,143 | 2,328 | 6,115 |

CYC:

| | $t = 1.0$ | | $t = 1.2$ | | $t = 1.5$ | | $t = 2.0$ | | $t = 3.0$ | | $t = 5.0$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $d = 1$ | 29 | 1.17 | 35 | 1.09 | 41 | 1.05 | 47 | 1.02 | 60 | 1.02 | 72 | 1.01 |
| $d = 2$ | 142 | 1.25 | 287 | 1.07 | 442 | 1.03 | 607 | 1.01 | 1027 | 1.00 | 1476 | 1.00 |
| $d = 3$ | 505 | 1.32 | 937 | 1.13 | 1451 | 1.07 | 2484 | 1.02 | 5311 | 1.01 | 10482 | 1.01 |
| $d = 4$ | 1784 | 1.41 | 3232 | 1.20 | 5716 | 1.10 | 11603 | 1.02 | 29963 | 1.01 | 69015 | 1.01 |
| $d = 5$ | 4432 | 1.57 | 8870 | 1.27 | 16806 | 1.13 | 37599 | 1.03 | 110186 | 1.02 | 249192 | 1.04 |
| $d = 7$ | 10698 | 2.16 | 25607 | 1.50 | 56337 | 1.21 | 150277 | 1.06 | 431875 | 1.09 | 832935 | 1.10 |
| $d = \infty$ | 36356 | 28.37 | 495360 | 3.33 | 1310965 | 1.34 | 1562064 | 1.20 | 1822427 | 1.12 | 2057597 | 1.07 |

Sumo:

| $d\backslash t$ | 1.0 | 1.2 | 1.5 | 2.0 | 3.0 | 5.0 |
|---|---|---|---|---|---|---|
| 1 | 4903 | 4911 | 4921 | 4936 | 4973 | 5038 |
| 2 | 5296 | 5395 | 5553 | 5823 | 6427 | 7743 |
| 3 | 6118 | 6451 | 7068 | 8280 | 10841 | 16337 |
| 4 | 6893 | 7556 | 9001 | 12176 | 18300 | 28878 |
| 5 | 7432 | 8517 | 11165 | 16945 | 26842 | 37284 |
| 7 | 7897 | 9991 | 15788 | 26203 | 36507 | 41443 |
| $\infty$ | 8047 | 15987 | 28353 | 35345 | 39389 | 41762 |

Mizar:

| $d\backslash t$ | 1.0 | 1.2 | 1.5 | 2.0 | 3.0 | 5.0 |
|---|---|---|---|---|---|---|
| 1 | 12 | 13 | 14 | 16 | 21 | 28 |
| 2 | 70 | 82 | 115 | 158 | 272 | 654 |
| 3 | 188 | 230 | 372 | 762 | 1950 | 5980 |
| 4 | 316 | 470 | 942 | 3021 | 8720 | 23440 |
| 5 | 540 | 979 | 2417 | 8179 | 22644 | 52241 |
| 7 | 1027 | 2708 | 8517 | 24445 | 54958 | 97481 |
| $\infty$ | 1116 | 8361 | 26959 | 57322 | 82379 | 107926 |

# SInE in E

How is SInE implemented in an actual theorem prover? E prover:

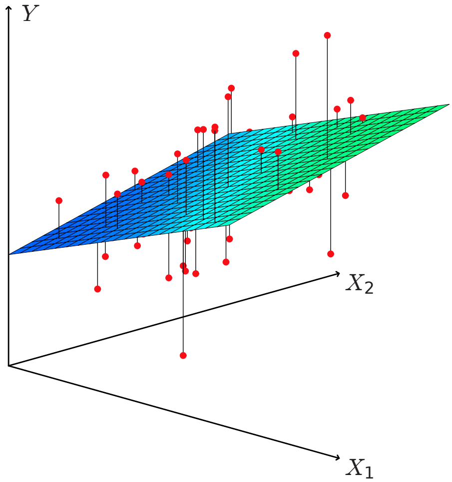
## Implementation: GSInE (in e_axfilter)

- Parameterizable filters
  - Different generality measures (frequency count, generosity, benevolence)
  - Different limits (absolute/relative size, # of iterations)
  - Different seeds (conjecture/hypotheses)
- Efficient implementation
  - E data types and libraries
  - Indexing (symbol $\rightarrow$ formula, formula $\rightarrow$ symbol)
- Multi-filter support
  - Parse & index once (amortize costs)
  - Apply different independent filters

## Primary use

- Initial over-approximation
  (efficiently reduce huge inputs to manageable size)
- Secondary use: Filtering for individual E strategies

# Linear Regression in Theorem Proving



# Linear Regression in Theorem Proving

## Premises: Classification

- Dimensions in the input
- Matrix QR decomposition

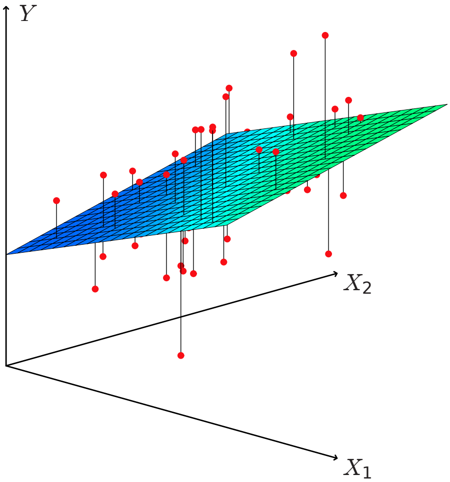
## Probabilities: Logistic

## Non-linearity

- Kernels *[Enigma]*
- Multi-output Ranking *[Kühlwein'14, ...]*

## State space reduction

- Random projections *[VowpalWabbit]*
- Decomposition



# Summary

## This Lecture

- Syntactic methods for premise selection
- Linear Regression

## Next

- decision trees
- deep learning in theorem proving

# Work Here / Homework

## Feature Space

- Assume each symbol corresponds to a feature. What algorithms support such an unlimited space?
- Propose (implement) a way to reduce the space to an algorithm that does not.

## Read Miller's HOL:

`http://www.lix.polytechnique.fr/Labo/Dale.Miller/papers/encyclopedia.pdf`