Machine Learning on Knowledge Bases

Marcus Pfeiffer

19th June 2018

Introduction

Data Preprocessing

Machine Learning Algorithms for Premise Selection

Premise Selectors in Detail

Comparison and Outlook

Introduction

Knowledge Bases and Provers

- Knowledge Bases

Knowledge Bases and Provers

- Knowledge Bases
  - Mizar Mathematical Library (MML)

Knowledge Bases and Provers

- Knowledge Bases
  - Mizar Mathematical Library (MML)
  - Library of Isabelle/HOL

Knowledge Bases and Provers

- Knowledge Bases
  - Mizar Mathematical Library (MML)
  - Library of Isabelle/HOL
  - SUMO
  - Cyc

Knowledge Bases and Provers

- Knowledge Bases
  - Mizar Mathematical Library (MML)
  - Library of Isabelle/HOL
  - SUMO
  - Cyc
- Automatic Theorem Provers

Knowledge Bases and Provers

- Knowledge Bases
  - Mizar Mathematical Library (MML)
  - Library of Isabelle/HOL
  - SUMO
  - Cyc
- Automatic Theorem Provers
  - E
  - Vampire
  - CVC4
  - SPASS
  - Z3

Knowledge Bases and Provers

- Knowledge Bases
  - Mizar Mathematical Library (MML)
  - Library of Isabelle/HOL
  - SUMO
  - Cyc
- Automatic Theorem Provers
  - E
  - Vampire
  - CVC4
  - SPASS
  - Z3
- Integration of Automatic Provers in Interactive Provers

Premise Selection problem

Definition (Premise Selection problem)

- ATP A
- large number of premises $\mathcal{P}$
- conjecture c

Premise Selection problem

Definition (Premise Selection problem)

- ATP A
- large number of premises $\mathcal{P}$
- conjecture c

Predict those premises from $\mathcal{P}$ that are likely of use to A for constructing a proof for c

Introduction Example

$$\mathsf{map}\ f\ xs = ys \implies \mathsf{zip}(\mathsf{rev}\ xs)(\mathsf{rev}\ ys) = \mathsf{rev}(\mathsf{zip}\ xs\ ys)$$

Introduction Example

$$\text{map}\ f\ xs = ys \implies \text{zip}(\text{rev}\ xs)(\text{rev}\ ys) = \text{rev}(\text{zip}\ xs\ ys)$$

A proof can be found including the following two lemmas:

$$\text{length}\ xs = \text{length}\ ys \implies \text{zip}(\text{rev}\ xs)(\text{rev}\ ys) = \text{rev}(\text{zip}\ xs\ ys) \quad (1)$$

$$\text{length}(\text{map}\ f\ xs) = \text{length}\ xs \quad (2)$$

Introduction Example

$$\mathsf{used}[\,] = \mathsf{used}\ \mathit{evs}$$

Introduction Example

$$\text{used}[\,] = \text{used}\ evs$$

A straightforward proof uses the following lemmas:

$$\text{used}[\,] = \bigcup_B \text{parts}(\text{initState}\ B) \tag{3}$$

$$X \in \text{parts}(\text{initState}\ B) \Longrightarrow X \in \text{used}\ evs \tag{4}$$

$$(\bigwedge x.x \in A \Longrightarrow x \in B) \Longrightarrow A \subseteq B \tag{5}$$

$$b \in \bigcup_{x \in A} Bx \longleftrightarrow \exists x \in A.b \in Bx \tag{6}$$

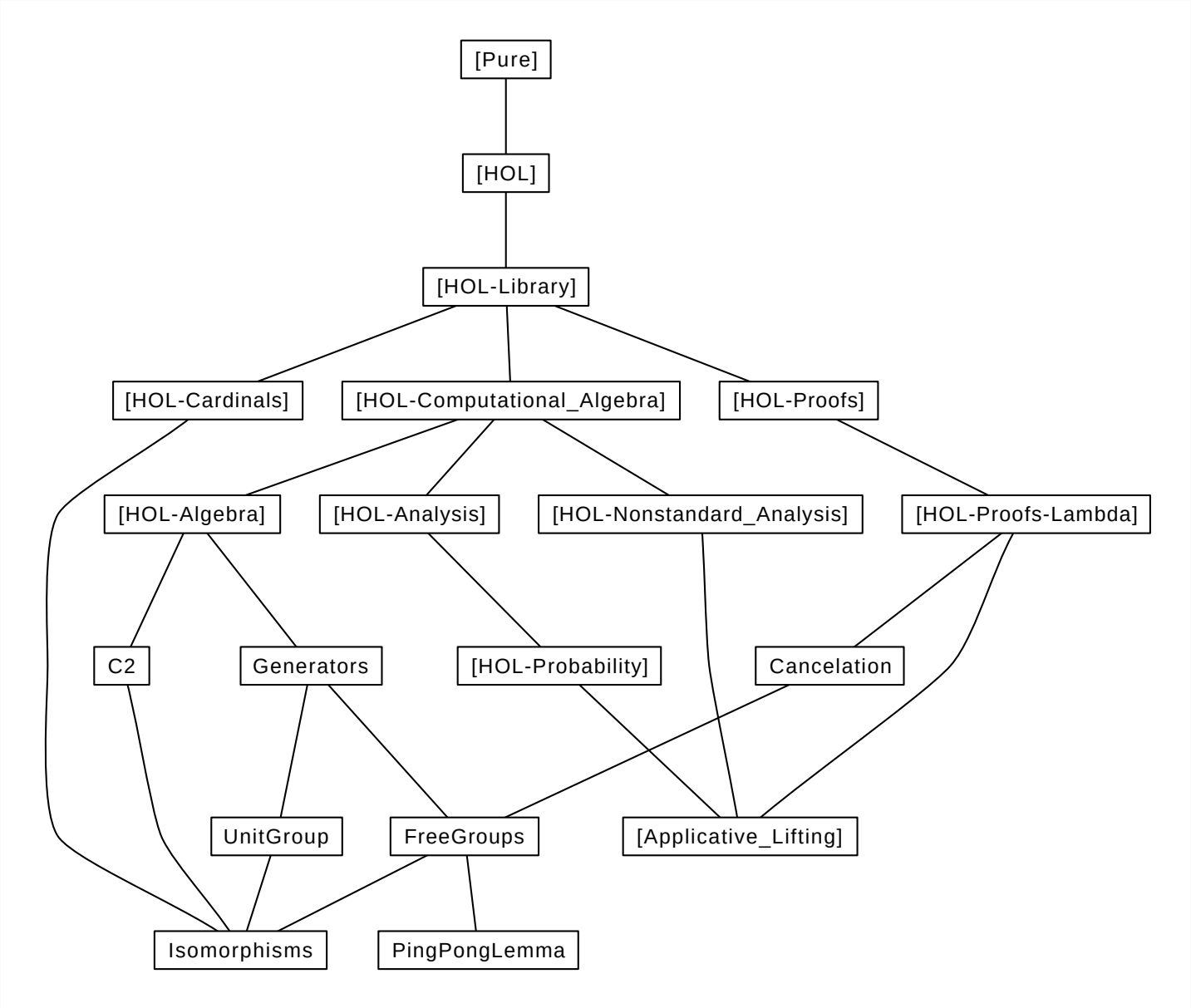
Data Preprocessing

Dependency

Definition (Dependency)

A definition or theorem T depends on some definition, lemma or theorem T', if T' is needed for the proof of T

Theory dependencies graph of Free Groups



Example of feature extraction

For given depth 2, $\boxed{g\,(h\,x\,a)}$ with $x :: \tau$ we get the following structural features:

Example of feature extraction

For given depth 2, $\boxed{g\,(h\,x\,a)}$ with $x :: \tau$ we get the following structural features:

x	a	$g\ \_$	$g\,(h\ \_\ \_)$
$h\ \_\ \_$	$h\,x\ \_$	$h\ \_\,a$	$h\,x\,a$

Example of feature extraction

For given depth 2, $\boxed{g\,(h\,x\,a)}$ with $x :: \tau$ we get the following structural features:

x	a	$g\ \_$	$g\,(h\ \_\ \_)$
$h\ \_\ \_$	$h\,x\ \_$	$h\ \_\ a$	$h\,x\,a$

which are simplified to

τ	a	g	$g(h)$
h	$h(\tau)$	$h(a)$	$h(\tau, a)$

Another example of feature extraction

transpose(map(map f) xss) = map(map f)(transpose xss)

has the features:

map
map(fun)
map(map)
map(transpose)
map(map, transpose)
map(list_list)
map(map, list list)
transpose
transpose(map)
transpose(list_list)
fun
list
list_list
List

Stored Information

$$(c, usedPremises(c), F(c))$$

Machine Learning Algorithms for Premise Selection

k-Nearest-Neighbors

"Nearness" of two formulas a and b in terms of shared features:

k-Nearest-Neighbors

"Nearness" of two formulas a and b in terms of shared features:

$$n(a, b) = \sum_{t \in F(a) \cap F(b)} w(t)^{\tau_1} \tag{7}$$

k-Nearest-Neighbors

"Nearness" of two formulas a and b in terms of shared features:

$$n(a, b) = \sum_{t \in F(a) \cap F(b)} w(t)^{\tau_1} \tag{7}$$

$N :=$ k nearest neighbors of c

k-Nearest-Neighbors

"Nearness" of two formulas a and b in terms of shared features:

$$n(a,b) = \sum_{t \in F(a) \cap F(b)} w(t)^{\tau_1} \tag{7}$$

$N :=$ k nearest neighbors of c

$$\text{Relevance}_c(p) =$$

$$\left(\tau_2 \sum_{q \in N \,|\, p \in usedPremises(q)} \frac{n(q,c)}{|usedPremises(q)|} \right) + \begin{cases} n(p,c) & \text{if } p \in N \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Naive Bayes

$$P(\text{p is used in the proof of c}) \tag{9}$$

Naive Bayes

$$P(\text{p is used in the proof of c}) \tag{9}$$

$$\approx P(\text{p is used to prove c'} \,|\, \text{c' has features F(c)}) \tag{10}$$

Naive Bayes

$$P(\text{p is used in the proof of c}) \tag{9}$$

$$\approx P(\text{p is used to prove c'} \,|\, \text{c' has features F(c)}) \tag{10}$$

$$\begin{aligned} & P(\text{p is used in the proof of c'}) \\ \cdot & \prod_{t \in F(c) \cap \bar{F}(p)} P(\text{c' has feature t} \,|\, \text{p is used in the proof of c'}) \\ \cdot & \prod_{t \in F(c) - \bar{F}(p)} P(\text{p has feature t} \,|\, \text{p is not used in the proof of c'}) \\ \cdot & \prod_{t \in \bar{F}(p) - F(c)} P(\text{c' does not have feature t} \,|\, \text{t is used in the proof of c'}) \end{aligned} \tag{11}$$

Naive Bayes

- $r(q)$ = number of times a fact q occurs as a dependency

Naive Bayes

- $r(q)$ = number of times a fact q occurs as a dependency
- $s(q, t)$ = number of times a fact q occurs as a dependency of a fact described by feature t.

Naive Bayes

- $r(q)$ = number of times a fact q occurs as a dependency
- $s(q, t)$ = number of times a fact q occurs as a dependency of a fact described by feature t.
- K = total number of known proofs.

Naive Bayes

- $r(q)$ = number of times a fact q occurs as a dependency
- $s(q, t)$ = number of times a fact q occurs as a dependency of a fact described by feature t.
- K = total number of known proofs.

Naive Bayes

- $r(q)$ = number of times a fact q occurs as a dependency
- $s(q, t)$ = number of times a fact q occurs as a dependency of a fact described by feature t.
- K = total number of known proofs.

$$P(\text{p is used in the proof of c'}) = \frac{r(p)}{K} \tag{12}$$

Naive Bayes

- $r(q)$ = number of times a fact q occurs as a dependency
- $s(q,t)$ = number of times a fact q occurs as a dependency of a fact described by feature t.
- K = total number of known proofs.

$$P(\text{p is used in the proof of c'}) = \frac{r(p)}{K} \tag{12}$$

$$P(\text{c' has feature t} \,|\, \text{p is used in the proof of c'}) = \frac{s(p,t)}{r(p)} \tag{13}$$

Naive Bayes

- $r(q)$ = number of times a fact q occurs as a dependency
- $s(q,t)$ = number of times a fact q occurs as a dependency of a fact described with feature t.
- K = total number of known proofs.

$$\text{Relevance}_c(p) = \sigma_1 \ln(r(p)) + \sum_{t \in F(c') \cap \bar{F}(p)} w(f) \ln \frac{\sigma_2 s(p,t)}{r(p)}$$
$$+ \sigma_3 \sum_{t \in \bar{F}(p) - F(c)} w(f) \ln \left(1 - \frac{s(p,t)}{r(t)}\right) + \sigma_4 \sum_{t \in F(c) - \bar{F}(p)} w(f)$$

Premise Selectors in Detail

MePo

For a set K of known symbols let

k = number of known symbols occurring in the fact

u = number of unknown symbols occurring in the fact

MePo

For a set K of known symbols let
k = number of known symbols occurring in the fact
u = number of unknown symbols occurring in the fact

1. For each fact, compute as its score $k/(k+u)$
2. Select the facts with the highest score;
   add the features of the selected facts to the set of known symbols K.

MaSh, MeSh

- MaSh: Machine Learning for Sledgehammer with k-NN and Naive Bayes

MaSh, MeSh

- MaSh: Machine Learning for Sledgehammer with k-NN and Naive Bayes
- MeSh: Combination of MaSh and a MePo like selector

MaSh, MeSh

- MaSh: Machine Learning for Sledgehammer with k-NN and Naive Bayes
- MeSh: Combination of MaSh and a MePo like selector
- In practice: Combined with a proximity selector

MaSh, MeSh

- MaSh: Machine Learning for Sledgehammer with k-NN and Naive Bayes
- MeSh: Combination of MaSh and a MePo like selector
- In practice: Combined with a proximity selector

Comparison and Outlook

Metrics

for $p_1, ...p_n$ selected by selection algorithm

Full recall:

$$\min_{k \in N} \; usedPremises(c) \subset \{p_1, ..., p_k\}$$

Metrics

for $p_1, ...p_n$ selected by selection algorithm

Full recall:

$$\min_{k \in N} \; usedPremises(c) \subset \{p_1, ..., p_k\}$$

k-Coverage

$$\frac{|\{p_1, ..., p_k\} \cap usedPremises(c)|}{min\{k, |usedPremises(c)|\}}$$

Comparison in Metric

		MaSh		MeSh	
Formalization	MePo	NB	kNN	NB	kNN
Auth	647	104	143	**96**	112
IsaFoR	1332	**513**	604	517	570
Jinja	839	244	306	**229**	256
List	1083	**234**	263	259	271
Nominal2	1045	**220**	276	229	264
Probability	1324	424	422	**393**	395

Figure 1: Average full recall

Comparison in Metric

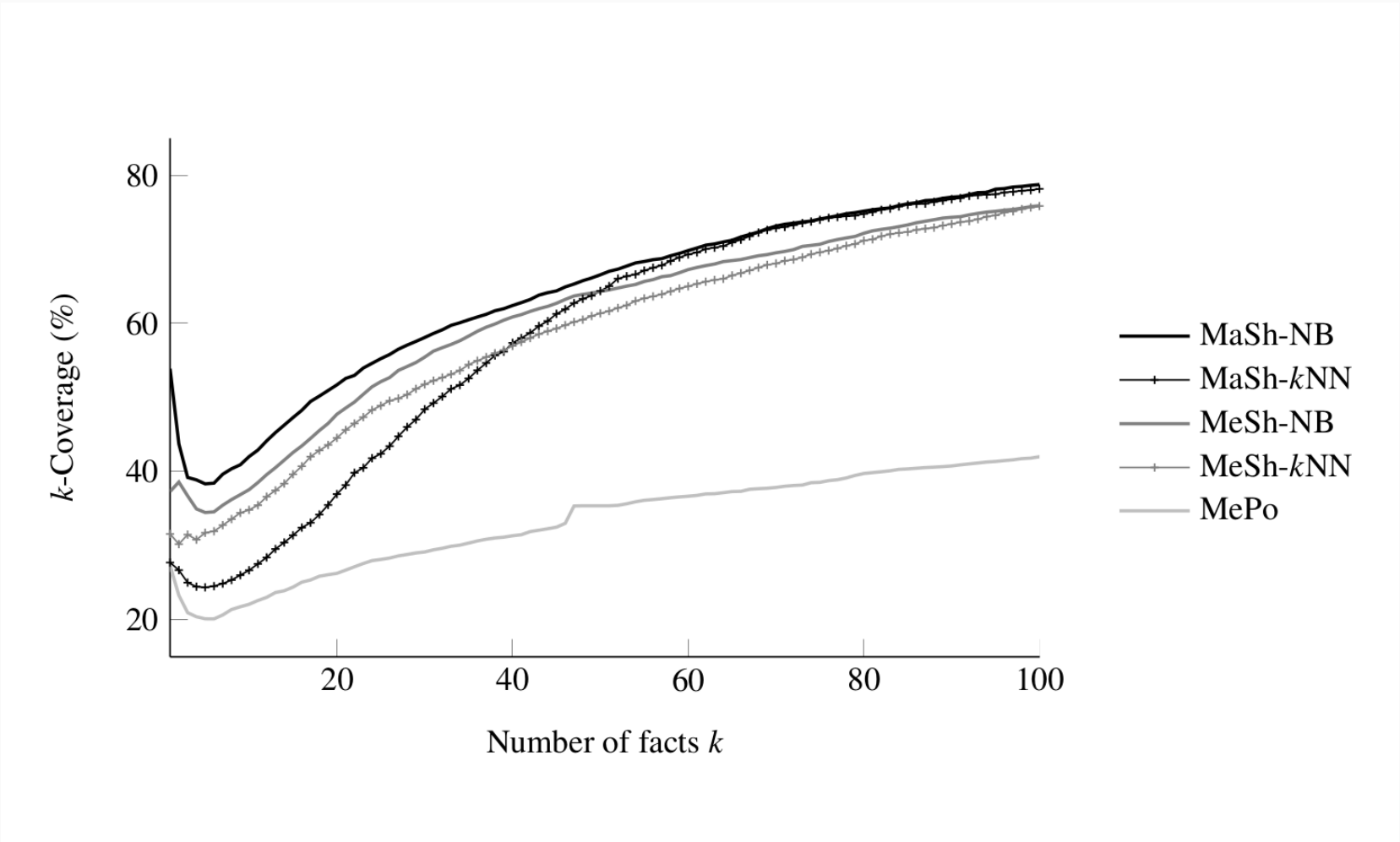


Figure 2: k-Coverage for IsaFoR

Comparison in Performance

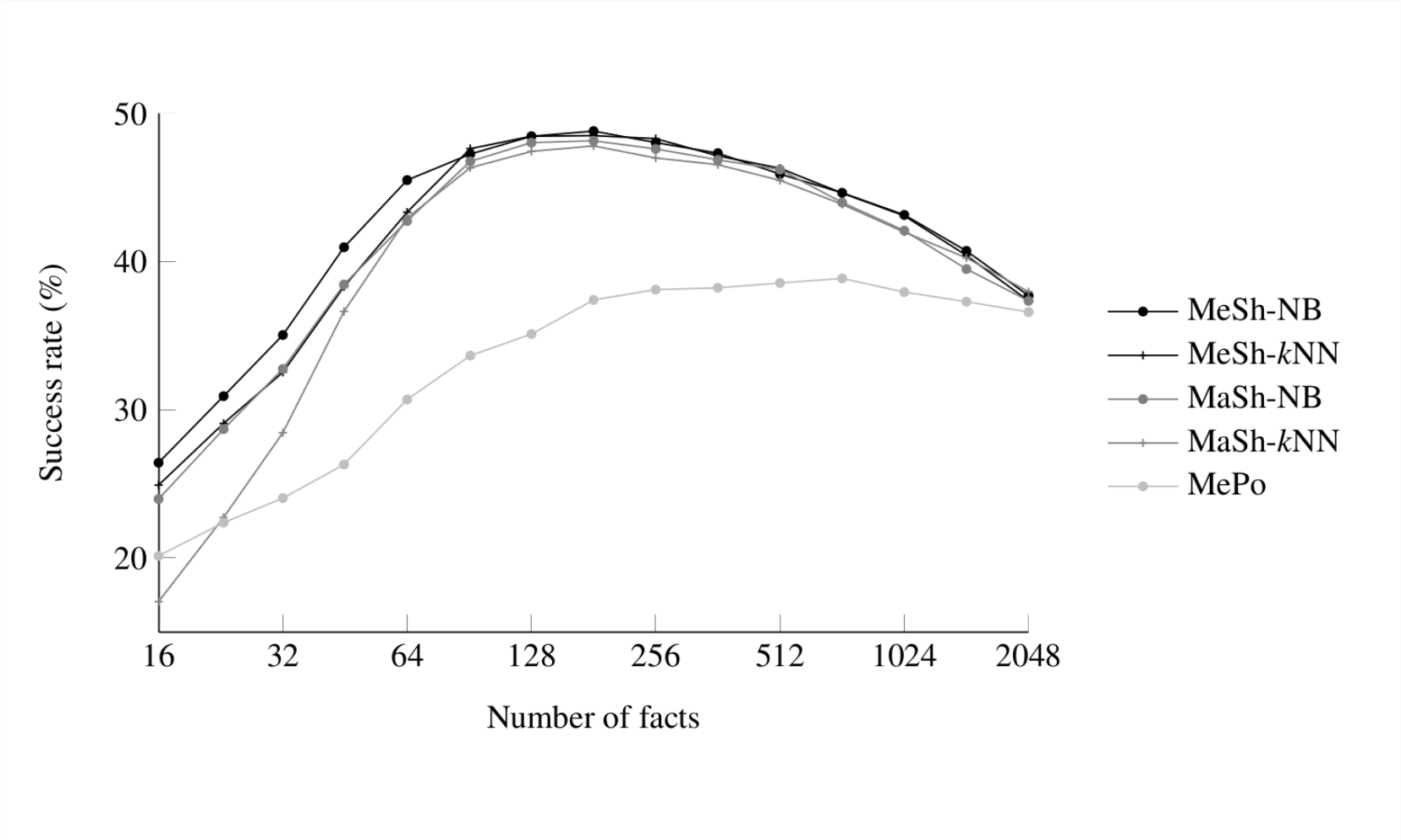


Figure 3: Success rates

Summary and Outlook

- Fact Selection is an important problem in Automatic Theorem Proving

Summary and Outlook

- Fact Selection is an important problem in Automatic Theorem Proving
- MePo performs well on problems with rare symbols

Summary and Outlook

- Fact Selection is an important problem in Automatic Theorem Proving
- MePo performs well on problems with rare symbols
- Fine grained dependency analysis and good feature extraction enable the successful application of machine learning algorithms

Summary and Outlook

- Fact Selection is an important problem in Automatic Theorem Proving
- MePo performs well on problems with rare symbols
- Fine grained dependency analysis and good feature extraction enable the successful application of machine learning algorithms
- MaSh and MeSh outperform MePo in terms of the chosen metrics

Summary and Outlook

- Fact Selection is an important problem in Automatic Theorem Proving
- MePo performs well on problems with rare symbols
- Fine grained dependency analysis and good feature extraction enable the successful application of machine learning algorithms
- MaSh and MeSh outperform MePo in terms of the chosen metrics
- Similar results for SNoW and MOR compared to Vampire-SInE for MML

Summary and Outlook

- Fact Selection is an important problem in Automatic Theorem Proving
- MePo performs well on problems with rare symbols
- Fine grained dependency analysis and good feature extraction enable the successful application of machine learning algorithms
- MaSh and MeSh outperform MePo in terms of the chosen metrics
- Similar results for SNoW and MOR compared to Vampire-SInE for MML
- Possibilities of applying other machine learning algorithms, e.g. Neural Networks